Brian Hanna
Meteor IP 2007
Microcontroller

MSP430 Overview:

The purpose of the microcontroller is to execute a series of commands in a loop while waiting for commands from ground control to do otherwise. While it has not received a command it populates data from the sensors and GPS to form packets of data which are regularly sent to the two meter radio for transmission via the APRS network and to the OSD for text overlay on the analog video signal that the platform will be transmitting. This requires the microcontroller to be able to control these systems. This is done through a series of control signals like a three bit signal, which gets sent to the video MUX which is then decoded, to tell the video MUX which camera should be outputting video and sending power to the corresponding camera.

Problems:

This design is a continuation of the previous teams design. The working parts of it will be reused and redesigns will be done as needed based on problems with the design that are discovered. One of the major problems that are considered is that the MSP430 outputs a logical signal with an absolute maximum of 3.3V and previously one of these signals was used by the video MUX board in an inverter chip that required a minimum of 4V for a logical high level in the input. To remedy this the inverter is going to be removed from the design and instead an extra control signal will be used so that there are two enables for the video MUX chips and two signals to select the channel. Also instead of decoding three bits in to signals used to power the camera that is currently selected to be outputting video the eight signals will be generated through code. In addition to the hardware change it will be necessary to update the software to reflect this change.

Requirements:

Some of the key requirements for the microcontroller include being able to operate within the expected temperature range, being able to control the cameras, being able to regularly generate packets of data and send them to the two meter radio for transmission and the OSD for text overlay, not being susceptible to hang-ups in the code. These are crucial because if these requirements are not met then after launching the platform the necessary data to retrieve the platform will not be received making the mission a complete failure. In order to prevent the processor getting stuck in the code the MSP430s watchdog timer will be used so that if a certain amount of time has passed and the processor has not finished executing the code then it will trigger an interrupt that will reset the system. Alternatively if the system receives a command from ground control to reset then it will execute the command.

Component Selection:

The MSP430 was chosen as the microcontroller because it was used in EE365 so it has the added benefit of familiarity in addition to meeting the basic needs of the platform without unnecessary complexity. It has enough ports to accommodate all the needed sensors. I2C is only available on USART0 so the digital sensors such as the temperature sensors, the accelerometer and the compass use this mode. USART0 was also used in UART mode to acquire data from the GPS and other sensors. USART1 stays in UART mode while it is listening for commands from ground control but after every minute it sends data to the radio it will be switched to SPI mode which will allow for data transmission to the SD card.

The EM-408 GPS was chosen because it is capable of operating within the expected temperature range. The data is received at 4800 baud and is used based upon two forms. The first is GPRMC which is used to acquire time, date, latitude, longitude, speed and heading data. It takes this form:

$GPRMC,005142.000,A,4304.7754,N,07741.1764,W,0.27,218.44,100407,,*1C

This data is defined in the following table.

Data	Meaning
005142.000	Time 00:51:42 UTC
A	Status A=active or void
4304.7754, N	Latitude 43 degree 04 minute 7754 sec N
07741.1764, W	Longitude 077 degree 41 minute 1764 sec W
0.27	Speed over ground in knots
218.44	Track angle in degree, true
100407	Data 10 April 07

The other form is GPGGA which is used to acquire the altitude. It takes this form:

$GPGGA,005143.000,4304.7752,N,07741.1763,W,1,08,1.1,169.3,M,-34.4,M,0000#64

where 169.3 represents an altitude of 169.3 meters.

Two crystals were selected, X415 6.5MHz and SE2405CT 32.768kHz, with different frequencies of oscillations because the RC like aspect of the system clock is very sensitive to temperature fluctuations and thus cannot deliver a reliable clock signal. The two different frequencies were used so that the processor can run at different speeds as needed.

The capacitors and resistors were chosen because they are within the expected temperature range the platform will experience during its ascent.

The TMP100 temperature sensors were chosen because they are both capable of operating within and measuring the temperatures of the expected range. They output 12 bits with a logical high value up to V_{DD} which requires a voltage divider to reduce it to a level that is usable to the MSP430 which receives the data and then does a calculation with the received bits to determine what the temperature is. They are digital sensors and utilize the I^2C bus and master-slave addressing. This facilitates having multipurpose purpose pins on the MSP430 so that the other digital sensors do not require their own pin to transmit their data to. Each digital sensor has its own slave address that it checks to see if it is equivalent to the data it is receiving after it has received a start signal.

Brian Hanna
Meteor IP 2007
Microcontroller

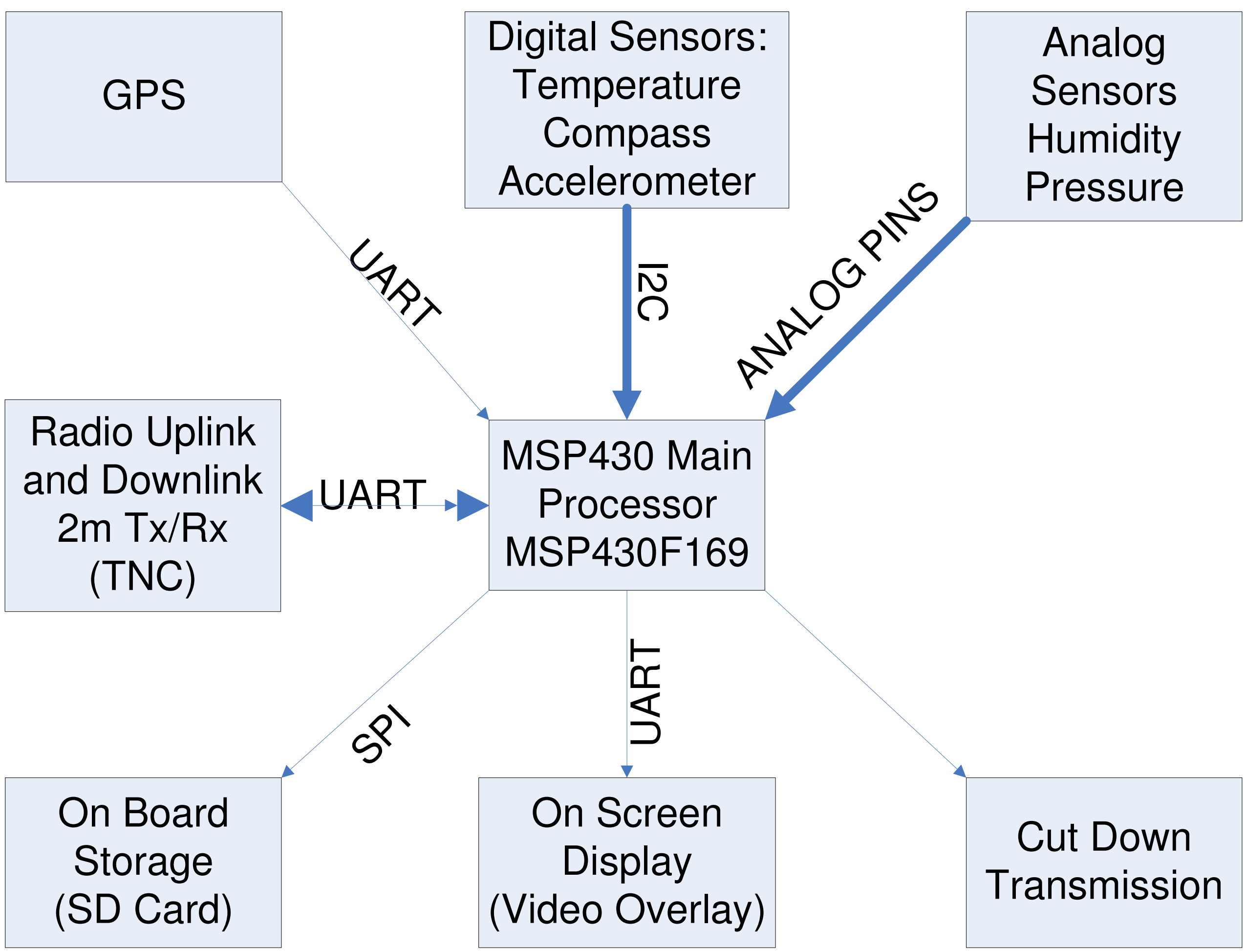
The HTM2500 relative humidity/temperature sensor was chosen because it is capable of operating within the expected temperature range.

The MPX series pressure sensors were chosen because they can operate within the expected temperature range. Two sensors have to be used however because the sensor that can measure higher pressures cannot measure pressure down to 0kPa so an additional sensor that could measure a range of pressures from 0kPa to 50kPa. When the pressure transitions below 20kPa then the processor starts taking data from the smaller sensor. This is needed because at 100,000ft the pressure is 0kPa.

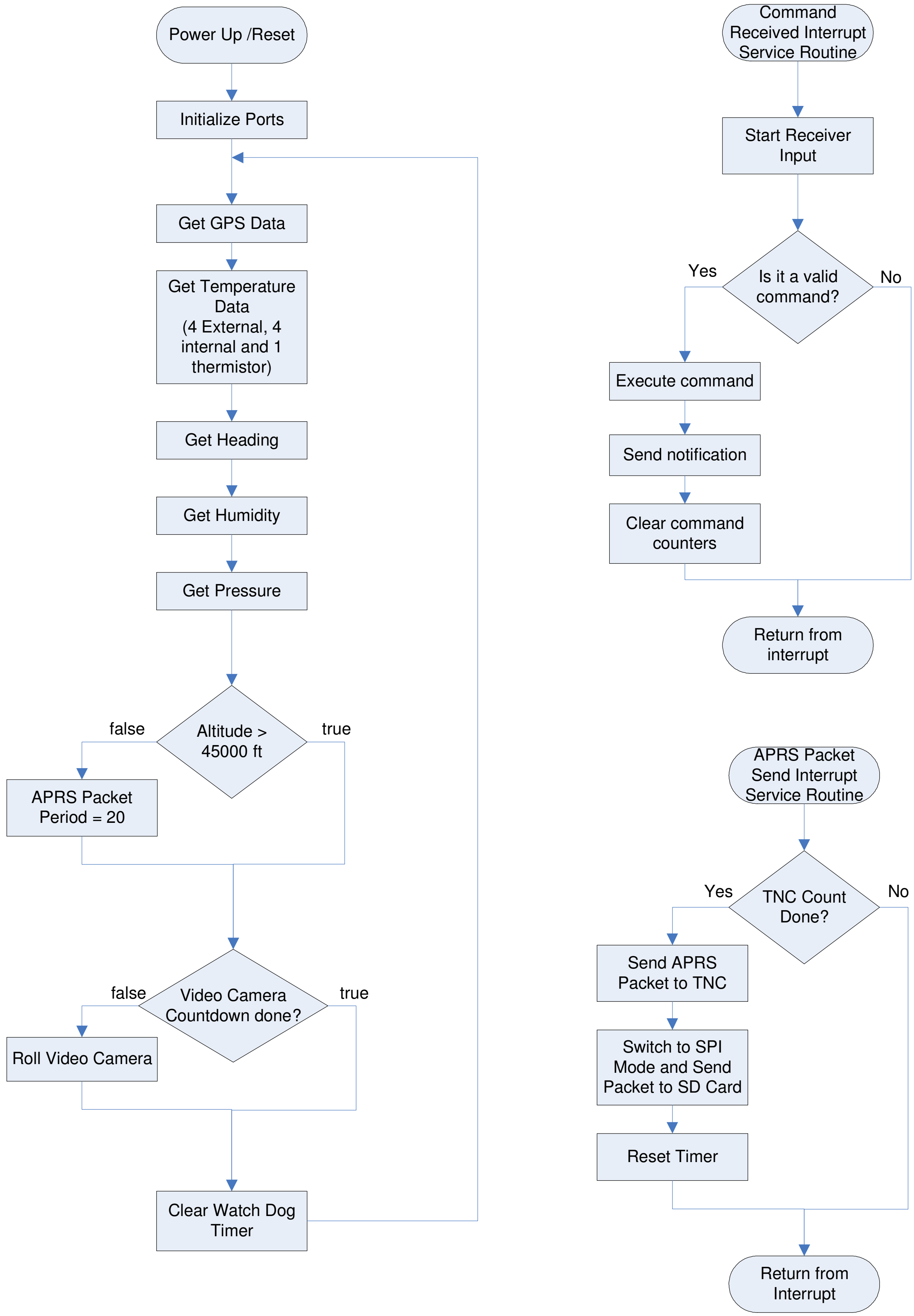
The B3F-1022 switch was chosen because it can operate within the expected temperature range. It is used as a manual reset for the system.

The MAX3232 RS232 receiver was chosen because it can operate within the expected temperature range. The TS3A5018 analog switch was chosen because it can operate within the expected temperature range. These two components work together so that in the debug phase of the project the MSP430 can output its packets through the analog switch which will be setup to decide that the RS232 should get the data or alternatively the RS232 can transmit a signal through the switch which the MSP430 receives and processes.

Subsystem Block Diagram:



Brian Hanna
Meteor IP 2007
Microcontroller
<u>Software Flow:</u>



Brian Hanna
Meteor IP 2007
Microcontroller
Instruction Set:

COMMAND LIST	DESCRIPTION
$COSDS	Recreate the OSD screen. Helpful if the OSD screen gets corrupted
$VCAM1	Choose the camera channel 1. Desired in order to see particular view.
$VCAM2	Choose the camera channel 2. Desired in order to see particular view.
$VCAM3	Choose the camera channel 3. Desired in order to see particular view.
$VCAM4	Choose the camera channel 4. Desired in order to see particular view.
$VCAM5	Choose the camera channel 5. Desired in order to see particular view.
$VCAMR	Roll camera channel to the next one.
$CUTDN	Cut down the platform from the zero pressure balloon
$APRSx	Change APRS packet period to 10x seconds
$TNCUx	Change APRS packet in TNC refresh period to 2x seconds
$CAMPx	Change Video Camera Channel Roll period to 20x seconds
$NAMES	Print names of the team members
$TESTS	Testing the Command Functionality
$FIRER	Fire the Rocket
$RESET	Resets the Microcontroller

Brian Hanna
Meteor IP 2007
Microcontroller

Test Plan:

In order to test that the controller can control the cameras a command can be sent to the controller to switch cameras and the voltage corresponding to the MUX enables, selects and power signals for the cameras can be measured to make sure it is responding as expected. In order to test that the controller can send data packets to the OSD as well as proper operation of the sensors a command can be sent to the controller to populate the data for the packet and by means of an RS232 port on a computer the output can be tested.